This listing of claims will replace all prior versions, and listings, of claims without prejudice in the application:

LISTING OF CLAIMS

1. (Previously presented) A method for implementing transaction services patterns, comprising the steps of:

(a) batching logically related requests for reducing network traffic, including the steps of managing a group of business objects necessary for a transaction in a logical unit of work, and grouping the logically related requests received from the logical unit of work into a single network message, wherein the logically related requests include at least a dependent batched request and a parent batched request;

(b) indicating whether the dependent batched request depends on the response to the parent batched request, including the steps of receiving a register that the dependent batched request is dependent upon response data from the parent batched request, receiving a response to the parent request, directing data from the response to the parent request to the dependent batched request; and receiving a response to the dependent batched request based on the response to the parent request;

(c) sending the single network message to the group of business objects necessary for the logical unit of work;

(d) sorting the logically related requests that are unbatched from a batched message; and

(e) providing multiple logical units of work operating concurrently, wherein the logical unit of work is one of the multiple logical units of work, such that each of the multiple logical units of work manipulates at least one of the group of business objects that is common to each of the multiple logical units of work, including the steps of creating a copy of the common business object for each of the logical units of work such that the copy of the common business object for each of the logical units of work is a separate instance of the common business object, and verifying

that a change to one instance of the common business object does not change the other copies of the common business object.

2. (Previously presented) A method as recited in claim 1, wherein the step of batching logically related requests further includes the steps of:

providing the group of business objects necessary for the transaction;

storing the single network message; and

sending the single network message upon receiving an order to send the message.

3. (Previously presented) A method as recited in claim 1, wherein the step of indicating whether the dependent batched request depends on the response to the parent batched request further includes the steps of:

providing the group of business objects necessary for the transaction;

sending the single network message across a network; and

unbundling the requests from the network message.

4. (Previously presented) A method as recited in claim 1, wherein the step of sending the single network message to the group of business objects includes the steps of:

providing the group of business objects necessary for the transaction;

creating a receiver which communicates with the business objects in the logical unit of work;

receiving a message for the business objects in the logical unit of work; and

directing the message to the receiver,

wherein the receiver forwards the message to each of the business objects in the logical unit of work.

5. (Previously presented) A method as recited in claim 1, wherein the step of sorting the logically related requests that are unbatched from a batched message includes the steps of:

providing the group of business objects necessary for the transaction;

obtaining at least one of sorting rules and sort weights;

sorting the requests in the message and placing them in a specific order determined from the one of the sorting rules and the sort weights;

batching the sorted requests into a single message;

sending the message to a data server; and

unbundling the requests from the message in the specific order.

6. (Previously presented) A method as recited in claim 1, wherein the step of providing multiple logical units of work operating concurrently further includes the steps of:

receiving a request to make changes to a copy of the business object of one of the logical units of work and changing that copy of the business object, wherein each copy of the business object knows the context of that copy of the business object in relation to the associated logical unit of work and wherein the other copies of the business object are not changed; and

updating the common business object based on the change to the copy of the business object.

7. (Previously presented) A computer program embodied on a computer readable medium for implementing transaction services patterns, comprising:

(a) a code segment that batches logically related requests for reducing network traffic, including a code segment that manages a group of business objects necessary for a transaction in a logical unit of work, and a code segment that groups the logically related requests received from the logical unit of work into a single network message, wherein the logically related requests include at least a dependent batched request and a parent batched request;

(b) a code segment that indicates whether the dependent batched request depends on the response to the parent batched request, including a code segment that receives a register that the dependent batched request is dependent upon response data from

the parent batched request, a code segment that receives a response to the parent request, a code segment that directs data from the response to the parent request to the dependent batched request; and a code segment that receives a response to the dependent batched request based on the response to the parent request;

(c) a code segment that sends the single network message to the group of business objects necessary for the logical unit of work;

(d) a code segment that sorts the logically related requests that are unbatched from a batched message; and

(e) a code segment that provides multiple logical units of work operating concurrently, wherein the logical unit of work is one of the multiple logical units of work, such that each of the multiple logical units of work manipulates at least one of the group of business objects that is common to each of the multiple logical units of work, including a code segment that creates a copy of the common business object for each of the logical units of work such that the copy of the common business object for each of the logical units of work is a separate instance of the common business object, and a code segment that verifies that a change to one instance of the common business object does not change the other copies of the common business object.

8. (Previously presented) A computer program as recited in claim 7, wherein the code segment that batches logically related requests further includes:

a code segment that provides the group of business objects necessary for the transaction;

a code segment that stores the single network message; and

a code segment that sends the single network message upon receiving an order to send the message.

9. (Previously presented) A computer program as recited in claim 7, wherein the code segment that indicates whether the dependent batched request depends on the response to the parent batched request further includes:

a code segment that provides the group of business objects necessary for the transaction;

a code segment that sends the single network message across a network; and

a code segment that unbundles the requests from the network message.

10. (Previously presented) A computer program as recited in claim 7, wherein the code segment that sends the single network message to the group of business objects includes:

a code segment that provides the group of business objects necessary for the transaction;

a code segment that creates a receiver which communicates with the business objects in the logical unit of work;

a code segment that receives a message for the business objects in the logical unit of work; and

a code segment that directs the message to the receiver,

wherein the receiver forwards the message to each of the business objects in the logical unit of work.

11. (Previously presented) A computer program as recited in claim 7, wherein the code segment that sorts the logically related requests that are unbatched from a batched message includes:

a code segment that provides the group of business objects necessary for the transaction;

a code segment that obtains at least one of sorting rules and sort weights;

a code segment that sorts the requests in the message and placing them in a specific order determined from the one of the sorting rules and the sort weights;

a code segment that batches the sorted requests into a single message;

a code segment that sends the message to a data server; and

a code segment that unbundles the requests from the message in the specific order.

12. (Previously presented) A computer program as recited in claim 7, wherein the code segment that provides multiple logical units of work operating concurrently further includes:

a code segment that receives a request to make changes to a copy of the business object of one of the logical units of work and changing that copy of the business object, wherein each copy of the business object knows the context of that copy of the business object in relation to the associated logical unit of work and wherein the other copies of the business object are not changed; and

a code segment that updates the common business object based on the change to the copy of the business object.

13. (Previously presented) A computer-readable storage medium containing a set of instructions for a general purpose computer for implementing transaction services patterns, the set of instructions comprising:

(a) means for batching logically related requests for reducing network traffic, including means for managing a group of business objects necessary for a transaction in a logical unit of work, and means for grouping the logically related requests received from the logical unit of work into a single network message, wherein the logically related requests include at least a dependent batched request and a parent batched request;

(b) means for indicating whether the dependent batched request depends on the response to the parent batched request, including means for receiving a register that the dependent batched request is dependent upon response data from the parent batched request, means for receiving a response to the parent request, means for directing data from the response to the parent request to the dependent batched request; and means for receiving a response to the dependent batched request based on the response to the parent request;

(c) means for sending the single network message to the group of business objects necessary for the logical unit of work;

(d) means for sorting the logically related requests that are unbatched from a batched message; and

(e) means for providing multiple logical units of work operating concurrently, wherein the logical unit of work is one of the multiple logical units of work, such that each of the multiple logical units of work manipulates at least one of the group of business objects that is common to each of the multiple logical units of work, including means for creating a copy of the common business object for each of the logical units of work such that the copy of the common business object for each of the logical units of work is a separate instance of the common business object, and means for verifying that a change to one instance of the common business object does not change the other copies of the common business object.

14. (Previously presented) A computer-readable storage medium as recited in claim 13, wherein the means for batching logically related requests further includes:

means for providing the group of business objects necessary for the transaction;

means for storing the single network message; and

means for sending the single network message upon receiving an order to send the message.

15. (Previously presented) A computer-readable storage medium as recited in claim 13, wherein the means for indicating whether the dependent batched request depends on the response to the parent batched request further includes:

means for providing the group of business objects necessary for the transaction;

means for sending the single network message across a network; and

means for unbundling the requests from the network message.

16. (Previously presented) A computer-readable storage medium as recited in claim 13, wherein the means for sending the single network message to the group of business objects includes:

means for providing the group of business objects necessary for the transaction;

means for creating a receiver which communicates with the business objects in the logical unit of work;

means for receiving a message for the business objects in the logical unit of work; and

means for directing the message to the receiver, wherein the receiver forwards the message to each of the business objects in the logical unit of work.

17. (Previously presented) A computer-readable storage medium as recited in claim 13, wherein the means for sorting the logically related requests that are unbatched from a batched message includes:

means for providing the group of business objects necessary for the transaction;

means for obtaining at least one of sorting rules and sort weights;

means for sorting the requests in the message and placing them in a specific order determined from the one of the sorting rules and the sort weights;

means for batching the sorted requests into a single message;

means for sending the message to a data server; and

means for unbundling the requests from the message in the specific order.

18. (Previously presented) A computer-readable storage medium as recited in claim 13, wherein the means for providing multiple logical units of work operating concurrently further includes:

means for receiving a request to make changes to a copy of the business object of one of the logical units of work and changing that copy of the business object, wherein each copy of the business object knows the context of that copy of the

business object in relation to the associated logical unit of work and wherein the other copies of the business object are not changed; and

means for updating the common business object based on the change to the copy of the business object.